Esperienze di programmazione
Lezione 5

Gianna Del Corso <delcorso@di.unipi.it>

18 Marzo 2015

1 Equalizzazione dell'istogramma

L'obiettivo dell'operazione di equalizzazione dell'istogramma è ottenere un'immagine risultato avente un'istogramma uniforme cioè "piatto". Questa caratteristica generalmente implica un' espansione della dinamica dei livelli di grigio e quindi un'incremento del contrasto. La regolarizzazione dell'istogramma potrebbe essere compiuta partendo dal modello continuo e arrivando al modello discreto.Nel caso del modello continuo avremmo denistà di distribuzione, nel caso discreto delle frequenze.

Per ottenere un istogramma piatto imponiamo che aree elementari dell'istogramma originale si trasformino in aree corrispondenti dell'istogramma modificato. Supponendo di lavorare nel continuo, sia $h(x)$ l'istogramma dell'immagine di partenza. Per realizzare l'equalizzazione è necessaria una trasformazione $T = T(x)$ tale che l'istogramma $g(T(x))$ dell'immagine trasformata sia costante, $g(T) = C$. Se si impone la condizione che le aree elementari dell'istogramma si trasformino in aree corrispondenti dell'istogramma modificato abbiamo:

$$h(x)dx = g(T)dT = CdT.$$

Vogliamo inoltre mantenere l'ordine dei livelli nella scala dei grigi, cioè

$$x_1 < x_2 \to T(x_1) < T(x_2)$$

in modo che la trasformazione sia monotona. Abbiamo quindi

$$\frac{dT}{dx} = \frac{1}{C}h(x),$$

e quindi $T(x) = \frac{1}{C}\int_0^x h(x)dx$. Nel discreto otteniamo

$$T(x) = \frac{1}{C}\sum_{k=0}^{x} h(k) = \frac{256}{M * N}\sum_{k=0}^{x} h(k) = 256\frac{\sum_{k=0}^{g}(h(k))}{\sum_{k=0}^{2} 55(h(k))},$$

dove C è definita notando che $\sum_{k=0}^{256} C = 256\,C = N * M$.

Generalizzando, definiamo la trasformazione del livello di grigio g attraverso la funzione $T(g)$ nel seguente modo

$$T(g) = 256 \frac{\sum_{k=0}^{g} (h(k))^m}{\sum_{k=0}^{255} (h(k))^m}. \tag{1}$$

Per $m = 1$ si parla di equalizzazione, per $m < 1$ di sottoequalizzazione e per $m > 1$di sovraequalizzazione.

Esercizio 1. Si scriva una funzione `IE=equalizza(I, m)` che restituisce l'immagine con l'istogramma equalizzato in accordo con la funzione (1). In particolare

- si costruica l'istogramma `h` di `I` chiamando la funzione `h=istogramma(I)`.
- Per ogni tono di grigio `g` costruisca la tabella `T` come definita in (1)
- Al pixel (i, j) di `IE` viene associato il tono di grigio ottenuto applicando la trasformazione T al tono di grigio presente nell'immagine originale `I`.

Si applichi la funzione a questa immagine, dopo aver fatto `I=imread('contrasto');` e `I=double(I)`. Si visualizzi poi con il comando `bar(hc)` l'istogramma `hc` dell'immagine equalizzata.

Si noti come l'immagine equalizzata poteva essere ottenuta semplicemente definendo la mappa di colore `map=T*[1 1 1]/256` e visualizzando l'immagine `I` con questa mappa.

2 Rotazione di un'immagine

E possibile ruotare una immagine attorno al pixel di coordinate (i_0, j_0) di un angolo α in senso antiorario semplicemente applicando una rotazione alle coordinate di ciascun pixel. Prima di vedere questo, osserviamo che nella nostra notazione matriciale la coppia (i, j) denota l'elemento della riga i e della colonna j. In questo modo l'origine degli assi è posta nell'angolo in alto a sinistra dell'immagine col primo asse che punta verso il basso e il secondo che punta a destra. Detto questo, ricordiamo che le coordinate del punto $P' = (x', y')$ ottenuto ruotando attorno a (x_0, y_0) di un angolo α in senso antiorario il punto P di coordinate (x, y) sono

$$\begin{aligned} x' - x_0 &= (x - x_0)\cos(\alpha) - (y - y_0)\sin(\alpha) \\ y' - y_0 &= (x - x_0)\sin(\alpha) - (y - y_0)\cos(\alpha). \end{aligned}$$

Quindi si ottiene

$$\begin{aligned} i' - i_0 &= (i - i_0)\cos(\alpha) - (j - i_0)\sin(\alpha) \\ j' - j_0 &= (i - i_0)\sin(\alpha) - (j - j_0)\cos(\alpha). \end{aligned}$$

Osserviamo che se (i, j) è una coppia di interi non è detto che (i', j') sia ancora una coppia di interi. Per meglio definire la funzione di rotazione denotiamo con

$A = (a_{i,j})$ la matrice corrispondente all'immagine di partenza, e con $B = (b_{i,j})$ la matrice corrispondente all'immagine ruotata. Allora, per eseguire la rotazione in modo più efficace, scandiamo tutte le coppie intere (i', j') corrispondenti ai pixel del supporto dell'immagine ruotata, applichiamo la trasformazione inversa ottenendo la coppia (i, j) non necessariamente di interi, ed andiamo a riempire l'elemento $b_{i',j'}$ con il valore du $a_{[i],[j]}$ dove $[i]$ e $[j]$ denotano gli arrotondamenti di i e j alla parte intera. Può accadere che la coppia $([i], [j])$ non stia nel supporto dell'immagine, cioè che $[i]$ non sia compreso tra 1 e n, e $[j]$ non sia compreso tra 1 e m. In tal caso assegnamo a $b_{i',j'}$ il valore 0. Scriviamoci qui sotto le formule inverse della rotazione cioè

$$\begin{aligned} i - i_0 &= (i' - i_0)\cos(\alpha) + (j' - i_0)\sin(\alpha) && (2) \\ j - j_0 &= -(i' - i_0)\sin(\alpha) + (j' - j_0)\cos(\alpha). && (3) \end{aligned}$$

Esercizio 2. Si scriva una funzione `function B=ruota(A, p, ang)` per ruotare l'immagine `A` attorno al pixel `p` di un angolo `ang` in senso antiorario. L'immagine restituita B si suppone della stessa dimensione dell'immagine di partenza. In particolare per ogni pixel (i', j') di B si va a vedere a che pixel (i, j) corrisponde in A utilizzando la (2) e se appartiene al suo supporto si pone $B_{i',j'} = A([i], [j])$. Si usi il comando `round(i)` per arrotondare all'intero gli indici i, j.

Esercizio 3. Come si può modificare la funzione precedente per far si che l'immagine ruotata non abbia gli angoli tagliati?

Esercizio 4. Come si può modificare la funzione `ruota(A, p, ang)` in modo da rimuovere i cicli for?